フォールト認知影響要因を考慮したソフトウェア信頼性モデリング枠組みに関する一考察 (不確実・不確定性下での意思決定過程)

AUTHOR(S):

井上, 真二; 山田, 茂

CITATION:

井上, 真二 ...[et al]. フォールト認知影響要因を考慮したソフトウェア信頼性モデリング枠組みに関する一考察 (不確実・不確定性下での意思決定過程). 数理解析研究所講究録 2010, 1682: 241-246

ISSUE DATE:

2010-04

URL:

http://hdl.handle.net/2433/141378

数理解析研究所講究録
第1682巻 2010年 241-246

フォールト認知影響要因を考慮したソフトウェア信頼性モデリング枠組みに関する一考察

鳥取大学・大学院工学研究科　井上 真二 (Shinji Inoue)†
鳥取大学・大学院工学研究科　山田　茂 (Shigeru Yamada)†
†Graduate School of Engineering, Tottori University

1　はじめに

ソフトウェア信頼度成長モデル（SRGM）は，ソフトウェア故障発生時間や所定の時間区間内におけるソフトウェア故障発生頻度もしくは発見されたフォールト数を確率量として，テスト工程もしくは運用段階におけるソフトウェア信頼度成長過程を記述する数理モデルである．これまでに数多くのSRGMが提案されているが，より現実的なモデルの開発を行うためには，実際のデバッギングプロセスを可能な限り忠実にモデルへと反映させたり，デバッギングプロセスに影響を与える要因を考慮するなどの方法が必要であろう．実際のテスト工程におけるデバッギングプロセスを考えた場合，ソフトウェア故障は，フォールトを含むあるプログラムパスが入力データによって実行された時に観測することができ，そのソフトウェア故障の原因であるフォールトを認知するためには原因解析の後，フォールトは除去される．また，このような作業を繰り返すことによってテスト工程におけるソフトウェアの信頼性は向上する．

このような実際のデバッギングプロセスを反映したSRGMは，これまでの研究においても数多く提案されている．Yamadaら [1] は，上述した現実的なプロセスに基づきながら遅延S字形SRGMと呼ばれる非同次ポアソン過程（NHPP）モデルを開発している．このSRGMは，デバッギングプロセスがソフトウェア故障発見過程とフォールト認知過程の2つのプロセスによって構成されるより現実的な仮定に基づいたSRGMである．近年では，上述したデバッギングプロセスを無限サーバ待ち行列モデルとして捉え，ソフトウェア故障の原因解析に要する時間を陽に考慮したソフトウェア信頼性評価のためのNHPPモデルに関する構築枠組みが提案されている [2,3]．このモデリングアプローチにおいて，現実的状況を反映しながら信頼性評価精度のさらなる向上を図るためには，比較的多くの労力を費やすソフトウェア故障の原因解析に要する時間について焦点をあて，より現実的な状況を陽にモデルへと反映させたSRGMの開発が必要である．一般的にフォールト認知過程においては，フォールト検出難易度，テスト技術者の能力，およびテスト人員などソフトウェア故障の原因解析時間に影響を与える様々なテスト環境要因が存在することが知られている [4]．本研究では，ソフトウェア故障の観測からフォールト除去に至るまでのプロセスをできるだけ忠実に表現するための1つのアプローチである無限サーバ待ち行列理論に基づくモデリング枠組みに沿って，これらの要因がフォールト認知作業時間に与える影響を考慮したSRGMの構築手法について議論する．

2　基本的モデリング枠組み [3]

ソフトウェア故障の観測からその原因となるフォールトの発見および除去に至るまでのプロセスは，ソフトウェア故障発生過程およびフォールト認知過程によって構成され，それぞれ相異なる意味をもつプロセスとして考えることができる．これら一連のプロセスにおいて発生する現象および事象を体系的かつ包括的に表現するためのアプローチとして，ソフトウェア信頼性評価のための無限サーバ待ち行列モデルが提案されている．このモデルは，以下の仮定に基づいて構築される：

(A-1)　ソフトウェア故障は，平均値関数 $H(t)$（強度関数 $h(t)$）をもつNHPPに従い観測される．
(A-2)　1つのソフトウェア故障が発生したとき，直ちにフォールト認知過程へと移り，その原因解析が行われた後，フォールトが発見される．
(A-3)　原因解析の開始からフォールトの修正・除去に至るまでの時間は，各々のソフトウェア故障に対して独立かつ同一の確率分布関数 $F(t)$ に従う．

いま，確率過程 $\{X(t), t \geq 0\}$ を時刻 t までに観測される総ソフトウェア故障数，$\{N(t), t \geq 0\}$ を時刻 t までに発見・修正される総フォールト数を表すものと定義する．このとき，時刻 $t = 0$ でテストが開始され

時刻 $t>0$ までに n 個のフォールトが発見・修正される確率は，

$$\Pr\{N(t)=n\} = \sum_{j=0}^{\infty} \Pr\{N(t)=n \mid X(t)=j\} \Pr\{X(t)=j\}$$
$$= \sum_{j=0}^{\infty} \Pr\{N(t)=n \mid X(t)=j\} \frac{[H(t)]^j}{j!} e^{-H(t)} \qquad (n \leq j) \tag{1}$$

として求められる．ここで，式 (1) の右辺における条件付き確率は，

$$\Pr\{N(t)=n \mid X(t)=j\} = \binom{j}{n} \{p(t)\}^n \{1-p(t)\}^{j-n}, \tag{2}$$

と与えられるものとする．式 (2) において，$p(t)$ は時刻 $x(<t)$ までに観測された 1 つのソフトウェア故障が時刻 t までにフォールト発見・修正に至る確率であり，

$$p(t) = \int_0^t F(t-x) \frac{dH(x)}{H(t)} \qquad (0<x<t), \tag{3}$$

として求められる．したがって，時刻 t までに発見された総フォールト数の確率関数は式 (2) および式 (3) を，式 (1) に代入することによって，

$$\Pr\{N(t)=n\} = \frac{\{M(t)\}^n}{n!} \exp[-M(t)] \tag{4}$$

として最終的に求められる．ここで，

$$M(t) = \int_0^t F(t-x) dH(x) \tag{5}$$

である．したがって，時刻 t までに発見された総フォールト数を表す確率過程 $N(t)$ は，本質的に，平均値関数 $M(t)$ をもつ NHPP に従うことになる．式 (5) に基づいて NHPP モデルを開発するためには，任意のテスト時刻 t までに観測される総期待ソフトウェア故障数を表す関数 $H(t)$ およびフォールト認知時間分布 $F(t)$ を具体的に与える必要がある．

3 テスト環境を考慮したフォールト認知時間分布と SRGM の構築

前述した一連のデバッギングプロセスの中でも，フォールト認知過程におけるソフトウェア故障の原因解析やフォールトの修正に要する時間は，フォールト検出難易度，テスト技術者の能力，テスト工数など様々な要因から影響を受けることが知られている [4]．本研究では，上述したモデリング枠組みにおいて，それらのフォールト認知影響要因を共変量とする比例ハザードモデル [5] に基づきながら，テスト環境要因を考慮したフォールト認知時間分布を構築すると共に，この手法に基づいたより現実的なモデリング枠組みの構築を行う．

3.1 モデリング枠組み

まず，$N(\geq 1)$ 組のフォールト発見数データと，それに対応したテスト時刻 $i(i=1,2,\cdots N)$ までの $q(\geq 1)$ 種類のテスト環境データ $z_{ij}(j=1,2,\cdots,q)$ を要素にもつ共変量ベクトル $\boldsymbol{Z}_i = (z_{1i}, z_{2i}, \cdots, z_{qi})$ が与えられているものとする．また，共変量 $\boldsymbol{Z}_i$ をもつ瞬間フォールト認知率を，

$$\lambda(t_i \mid \boldsymbol{Z}_i) = \lambda_0(t_i) r(\boldsymbol{Z}_i) \tag{6}$$

として与える．式 (6) において，$\lambda_0(t_i)$ は基本瞬間フォールト認知率，$r(\boldsymbol{Z}_i)$ は時間に依存しない相対的な瞬間フォールト認知率を表す関数であり，係数ベクトル $\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_q)$ を設定しながら，次のような対数線形性：$r(\boldsymbol{Z}_i) = \exp[\boldsymbol{\beta}^{\mathrm{T}} \boldsymbol{Z}_i] = \beta_1 z_{1i} + \beta_2 z_{2i} + \cdots + \beta_q z_{qi}$ を仮定する．ここで，T は転置行列を表す．これより，式 (6) は，

$$\lambda(t_i \mid \boldsymbol{Z}_i) = \lambda_0(t_i) \exp[\boldsymbol{\beta}^{\mathrm{T}} \boldsymbol{Z}_i] \tag{7}$$

と記述され，最終的にフォールト認知作業に影響を与える要因を考慮したフォールト認知時間分布は，

$$F(t \mid \boldsymbol{Z}_i) = \lambda(t \mid \boldsymbol{Z}_i) \exp\left[-\int_0^t \lambda(x \mid \boldsymbol{Z}_i)dx\right] \tag{8}$$

$$= \lambda(t \mid \boldsymbol{Z}_i)R(t) \tag{9}$$

のように求められる．また，このとき式 (5) の平均値関数は，

$$M(t) = \int_0^t H(t-x)\lambda(x \mid \boldsymbol{Z}_t)R(x)dx \tag{10}$$

となる．式 (10) から，ソフトウェア故障発見過程における平均的なソフトウェア故障観測数を表す関数 $H(t)$ とフォールト認知過程における基本瞬間フォールト認知率を表す $\lambda_0(t)$ に対して具体的かつ適切な関数を与えることによって，フォールト認知要因を考慮した NHPP モデルを構築することができる．

3.2 SRGM の構築

提案したモデリング枠組みに基づきながら，フォールト認知要因を考慮した NHPP モデルを具体的に構築する．まず，「モデル 1」として，ソフトウェア故障発見過程における平均的なソフトウェア故障観測数を表す関数に対してパラメータ b の指数形ソフトウェア信頼度成長モデル，および，フォールト認知過程における基本瞬間フォールト認知率に対して λ_0（一定）を仮定する，つまり，

$$H(t) = a(1 - \exp[-bt]), \tag{11}$$

$$\lambda(t \mid \boldsymbol{Z}_t) = \lambda_0 \exp[\boldsymbol{\beta}\boldsymbol{Z}_t](\equiv B) \quad (\lambda_0 > 0), \tag{12}$$

をそれぞれ仮定するとき．式 (10) の平均値関数は，

$$M(t) = a\left[1 - \frac{1}{b-B}(b\exp[-Bt]) - B\exp[-bt])\right] \tag{13}$$

のように求められる．ここで，$a(>0)$ はテスト開始前にソフトウェア内に潜在する総期待フォールト数，$b(>0)$ はソフトウェア故障発生率を表す．また，「モデル 2」として，

$$H(t) = a(1 - \exp[-bt]), \tag{14}$$

$$\lambda(t \mid \boldsymbol{Z}_t) = b\exp[\boldsymbol{\beta}\boldsymbol{Z}_t], \tag{15}$$

をそれぞれ仮定した場合，式 (10) の平均値関数は，

$$M(t) = a\left[1 - \frac{1}{1-S}(\exp[-btS] - \exp[-bt+S])\right] \tag{16}$$

のように求められる．ただし，$S = \exp[\boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{Z}_t]$ である．

4 ソフトウェア信頼性評価尺度

ソフトウェア信頼性評価尺度は，定量的な信頼性評価を行う上で有用な評価尺度として知られている．ここでは，代表的な以下の 3 つの信頼性評価尺度について簡単に議論する．まず，期待残存フォールト数は，任意のテスト時刻 t においてソフトウェア内に残存するフォールト数の期待値を表す尺度であり，確率過程 $\{N(t), t \leq 0\}$ が式 (4) の NHPP に従う場合，次のように導出される．

$$E(t) \equiv \mathrm{E}[\bar{N}(t)] = \mathrm{E}[N(\infty) - N(t)]$$

$$= M(\infty) - M(t). \tag{17}$$

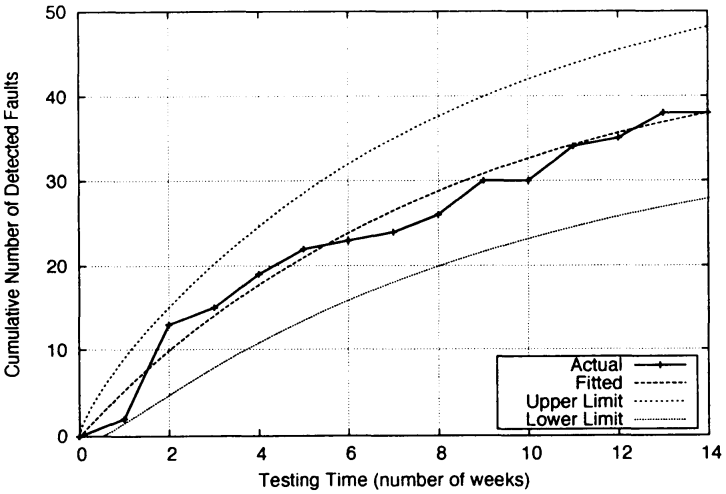

図 1： 推定された平均値関数 $\widehat{M}(t)$. (モデル 1)

ここで，$M(t)$ は NHPP の平均値関数を表す．また，ソフトウェア信頼度関数は，任意の時刻 t までテストが進行しているという条件の下で，その後の時間区間 $(t, t+x](t \geq 0, x \geq 0)$ においてソフトウェア故障が発生しない条件付き確率として定義される．したがって，ソフトウェア信頼度関数 $R(x \mid t)$ は，

$$R(x \mid t) \equiv \sum_{k} \Pr\{N(t+x) = k \mid N(t) = k\} \Pr\{N(t) = k\}$$
$$= \exp[-\{M(t+x) - M(t)\}] \tag{18}$$

のように導出される．最後に，累積 MTBF（mean time between software failures）は，ソフトウェア故障発生時間間隔分布が通常の確率分布関数の性質を満たさない場合における平均ソフトウェア故障発生時間間隔の代替的尺度の 1 つであり，

$$MTBF_C(t) = \frac{t}{M(t)} \tag{19}$$

として求められる．

5 適用例

今回適用する実測データは，14 組のフォールト発見数データ [6] であり，フォールト認知作業に影響を与えるテスト環境要因としてフォールト発見工数（人時，z_1）およびフォールト発見のための計算機時間（hr，z_2）を使用する．このとき，モデル 1 と名付けた NHPP の平均値関数である式 (13) に含まれるパラメータ a，b，λ_0，β_1，および β_2 は最尤法を用いて同時に推定する．これらのパラメータを最尤法を用いて同時に推定した結果，パラメータ推定値：$\widehat{a} = 46.861$，$\widehat{b} = 0.1190$，$\lambda_0 = 1.6398$，$\widehat{\beta_1} = 0.9227$，および $\widehat{\beta_2} = -4.0350$ をそれぞれ得た．図 1 に，これらのパラメータ推定値を用いて推定された平均値関数を示す．図 1 より，推定された平均値関数は実測データが示す挙動をある程度表現できていることがわかる．

次に，推定された種々のソフトウェア信頼性評価尺度について議論する．図 2 に，推定された期待残存フォールト数の時間的挙動とパラメータ推定値 $\widehat{a}$ をテスト開始前に潜在する総フォールト数として取り扱ったときの実測データに基づく挙動を示している．図 2 から，テスト終了時刻 $t = 14$（週）における総期待残存フォールト数 $\widehat{E}(14)$ は 8.8612（約 9 個）と推定される．また，図 3 および図 4 に，それぞれ，推定されたソフトウェア信頼度関数 $\widehat{R}(x \mid 14)$ および $\widehat{MTBF}(t)$ の挙動を示す．当該ソフトウェアがテスト開始後 14 週目にリリースされ運用段階においてもテスト工程と同様の環境で使用した場合，図 3 から，リリース後 1 週目におけるソフトウェア信頼度は約 0.3699 と推定される．さらに，図 4 より，テスト終了時刻における累積 MTBF は $\widehat{MTBF}_C(14) \approx 0.3684$（週）（約 61.9 時間）と推定される．

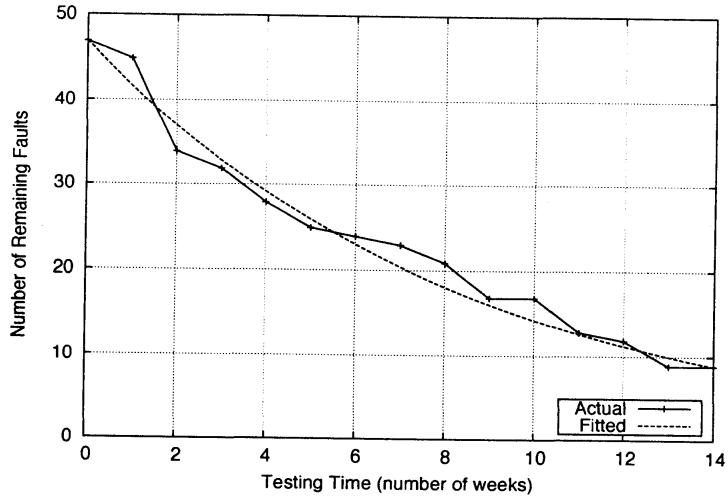

図 2： 推定された期待残存フォールト数 $\widehat{E}(t)$. (モデル 1)

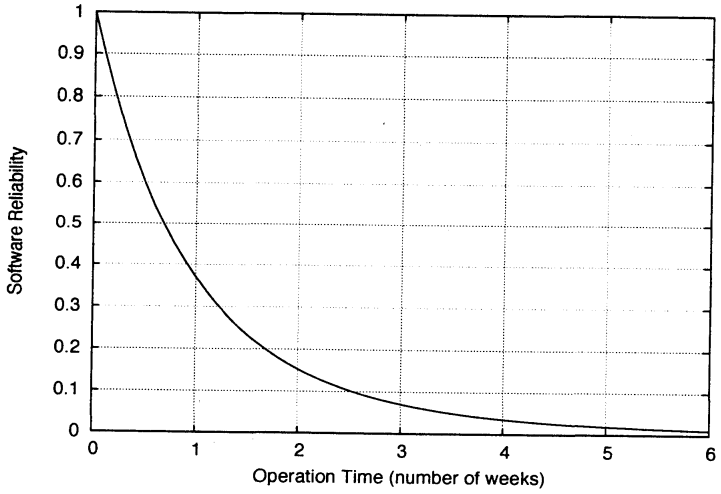

図 3： 推定されたソフトウェア信頼度関数 $\widehat{R}(x \mid 14)$. (モデル 1)

6 おわりに

本研究では，フォールト認知作業に影響を与える要因が存在することを考慮しながら，すでに提案されている無限サーバ待ち行列モデルに基づいたソフトウェア信頼度成長モデリング枠組みの拡張を行った．具体的には，比例ハザードモデルの考え方に基づき，種々のテスト環境要因がフォールト認知作業に与える影響を陽に反映しながら，ソフトウェア信頼性評価のための無限サーバ待ち行列モデルにおけるフォールト認知時間分布の拡張を行った．また，拡張されたフォールト認知時間分布をソフトウェア信頼性評価のための無限サーバ待ち行列モデルに適用した場合に導出できるモデリング枠組みについても議論した．最後に，議論したモデリング枠組みに基づいて具体的な 2 種類の SRGM を構築すると共に，その中の 1 つの SRGM について，実測データを用いたソフトウェア信頼性解析例を示した．

今回議論したモデリング枠組みは，ソフトウェア故障発見過程において観測された累積ソフトウェア故障数の平均的挙動を記述する関数とフォールト認知過程における基本瞬間フォールト認知率に対して，ある適切な関数を与えることで最終的にフォールト認知影響要因を考慮した SRGM を構築することができる．一方，議論した枠組み自体が複雑であるため，それぞれの過程において適切な関数をそれぞれ与えた場合でも，当該枠組みに基づいた SRGM を導出できない場合があるため，解析的な意味で重要な問題を孕んでい

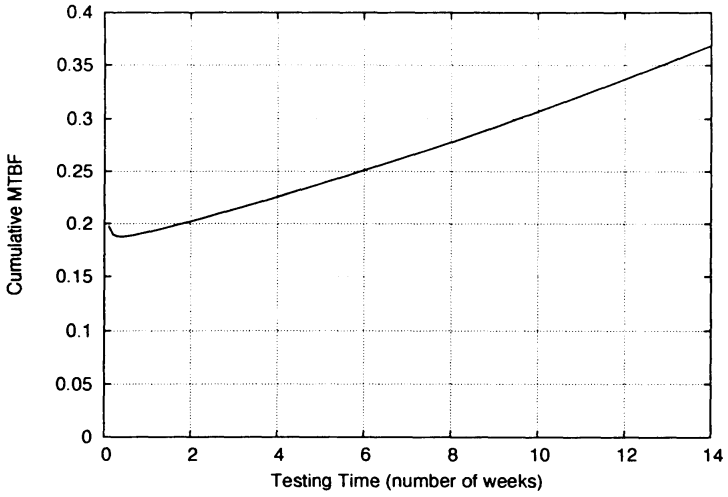

図 4： 推定された累積 MTBF$\widehat{MTBF}(t)$. (モデル 1)

る．また，多くのフォールト認知影響要因を採用した場合，すなわち，共変量ベクトルの要素が多い場合，推定すべきパラメータが多くなるため，尤度方程式を数値的に解くようなパラメータ推定手法では限界がある．このように今回議論したモデリング枠組みは，実用に際して重要な問題を有しているが，今回の研究において議論したモデリング枠組みに基づきながら，今回構築した SRGM 以外のモデルを数多く構築して，提案手法の有効性を検証する必要がある．

参考文献

[1] S. Yamada, M. Ohba, and S. Osaki, "S-shaped reliability growth modeling for software error detection," *IEEE Trans. Reliab.*, **R-32** (5), pp. 475–478, 484, 1983.

[2] T. Dohi, T. Matsuoka, and S. Osaki, "An infinite server queueing model for assessment of the software reliability," *Elec. Commu. Japan*, Part III, **85** (3), pp. 43–51, 2002.

[3] S. Inoue and S. Yamada, "An extended delayed S-shaped software reliability growth model based on infinite server queueing theory," in *Rel. Model., Ana. Opti.*, H. Pham, Ed., pp. 357–372, World Scientific, Singapore, 2006.

[4] C.Y. Huang and W.C. Huang, "Software reliability analysis and measurement using finite and infinite server queueing models," *IEEE Trans. Reliab.*, **57** (1), pp. 192–203, 2008.

[5] 中村剛，Cox 比例ハザードモデル，朝倉書店，2001.

[6] K. Rinsaka, K. Shibata, and T. Dohi, "Proportional intensity-based software reliability modeling with time-dependent metrics," *Proc. COMPSAC'06*, pp. 369–376, 2006.